BEEBLET

ISSN 0112-3351

VOLUME 2 NUMBER 9 OCTOBER 1984

BBC/ACORN
COMPUTER USER GROUP
NZ (inc)

CONTENTS

Editorial 3
A letter to the Editor 4
A challenge 5
Another letter 7
Mailbag 8
Teletext Title Page 11
A plug for the Beeb 14
Relocating VIEW 18
Edit your own user defined characters 21
Structure diagrams 28
Files - Part 3 31
HELP 36
Hall of Fame 39
Local meetings... 40

ADVERTISER'S INDEX

Einstein Scientific..................35
FG Software..........................38
John Gilbert Electronics........... 6
Molymerx.............................27

BEEBLET is edited by Bruce Wills who may sometimes be tracked down on 861-662 (h) or 720-030 (wk).

DISCLAIMER

The views of the reviewers, the Editor and other contributors are not necessarily shared by the BBC/ACORN COMPUTER USER GROUP OF NZ INC.

EDITORIAL

This issue was particular tough to produce for some reason - possibly I had to type more of the items in than usual. Anyway, I hope that there will be a lot of reading in it for ALL members - you don't have to be a programming genius to make a very worthwhile contribution to Beeblet. I would like to see a lot more news about what people are doing, what problems they are having, what they have achieved. It would be nice to hear from a few more Electron owners too.

****

Anyone feel like organising a national convention? Need not be in one of the main centres, should be at a reasonably inexpensive location with provision for families. Any suggestions, starting with suitable sites, and timing?

****

The COMPETITION to produce educational software for young children continues - see the September issue for further details.

****

DISK ORDER

Jill Armstrong of the Wellington branch is keeping in close touch with the local agent re the bulk order. Evidently they have had someone in the States sorting out the problems - we are not the only ones affected. I expect that we will get some action before the next Beeblet (I am getting close to running out of disc space...).

****

I notice that another micro group is suggesting that different areas might investigate the prospect of billeting people from other areas on visits. We could certainly do with a bit more inter-branch communication.

****

Don't forget to check the ads in this issue: local retailers now carry a very good range. Some of them are wondering where all the BBC people have gone.

A LETTER TO THE EDITOR

Anne.

In recent UK computer magazines, the questions "where are the women" and "how can they be encouraged into computer use?" have been discussed. I would suggest that they are staying out of another field where women are depicted as being capable of only "amusing" men.

Cartoons and drawings stand out from text, and the one on page 31 of the September Beeblet was noticed, and unfavourably commented on by several women on a computer course I am involved in. As former Editor of this magazine, I did not put all my (female) efforts into producing it, only to have such drawings published in it as will alienate a section of its readership.

Furthermore, that particular drawing is unoriginal - I have seen the identical scenario hand drawn with different initials, and have also seen drawings in very similar vein in an American magazine.

P.S. How about getting some of the readership to design an interface for that marvellous "3D" printer on page 7? I'm sure it would out-do the "Beastie" currently available for the Beeb.

Editor's Comment:

Great! Someone has finally got around to telling me what they do/don't want to see in Beeblet. Those who feel alienated from the magazine's content have two options:

- do nothing
- tell me about it

Only one of these options is likely to result in any improvement.

The cartoon is in fact as original as any published cartoon is likely to be - it was sketched by Ricardo Delee some time ago and then computerised for publication in Beeblet by Tim Ryan. It is worth thinking about that cartoon - just what is it commenting on?

Anne made the point that graphics stand out from text. This is why I was interested in the cartoon: it broke up a great sea of text. It was also intended to suggest that contributions to Beeblet don't need to be highly sophisticated programs (which I suspect alienate some of the group).

A CHALLENGE!

Some of you will be familiar with the puzzles published in the English magazine New Scientist. A member of a Wellington micro club has obtained permission from New Scientist to use these, and I thought they might appeal to a few members of our group.

First, a warm up - ENIGMA is exactly divisable by EE NN II GG MM AA BB and CC. Each individual letter represents a different digit.

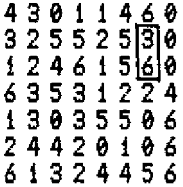
Which number does ENIGMA represent?

NB. There is ONE correct solution.

Your editor solved this one fairly quickly without computer assistance, using pure logic to reduce the possibilities down to about a dozen numbers. I then discovered a major flaw in my chain of deductions.....

Now something more difficult:

The matrix below represents a full set of 28 dominos (from 00 to 66) arranged in a 8x7 block. The position of the 63 domino is marked. Where are the boundaries between the other dominos?



```
4 3 0 1 1 4 6 0
3 2 5 5 2 5 3 0
1 2 4 6 1 5 6 0
6 3 5 3 1 2 2 4
1 3 0 3 5 5 0 6
2 4 4 2 0 1 0 6
6 1 3 2 4 4 5 6
```

Again, there is ONE correct solution.

Solutions should be accompanied by a detailed explanation and listing, if appropriate. The only reward offered is the intellectual satisfaction, and additional experience, derived from the exercise however the same problem is likely to be posed to the local Apple Users Group. People might like to note run times....

JOHN GILBERT ELECTRONICS

Parnell Rise, P.O. Box 37-245, AUCKLAND. Telephone (09) 30-839

Magazine	Per Copy	+ Postage	Annual Sub.
BEEBUG	$4-95	+ $0-50	$50-00
ELBUG	$4-95	+ $0-50	$50-00
ELECTRON USER	$4-95	+ $0-50	$50-00
EDUCATIONAL COMPUTING	$4.95	+ $0.50	$50-00
MICRO USER	$5-50	+ $0-75	$66-00

16K SOLIDISK Sideways RAM : $210 + sales tax

DATALIFE diskettes (5.25",SSDD,40T but useable 80T,) : $59-95 / 10 + $2-50 pp.

SPECIAL :- PACESETTER 10" dot matrix PRINTERS. CENTRONICS parallel, tractor / friction feed, 80 cps, Bidirectional, Superscript, Bit image graphics.

LIMITED QUANTITY : $750-00 + $5-00 freight.

NB. SCHOOLS : Special tax free prices for school purchases.

BBC SOFTWARE

$29-50:

SP -Strategy 1 -Strategy 2 -Detective
-Froglet -Demon Knight -Stockmarket
-Fall of Rome -White Burrows
Vision -Snooker
Imagine -Cosmic Cruiser
Ocean -Mr Wimpy
PSS -Invader -Sentinal -Light Cycle

$39-50:

-Planetfall -The Valley
Vision -Nifty Lifty -Digger

$27-00:

Superior -Hunchback -2002 -Crazy Painter
-Road Runner -Frogger

A LETTER

John B.   Waiheke Island

From one oldie to all you other oldies out there, get a computer, they are fun! If you feel that there is such a thing as a generation gap then reduce the gap by taking up computing. My young grandson aged 10 years tells his contempories (with some pride), "my grandad is into computing."

Yes, I bought a computer, an Electron. I did not know a thing about computing. I'm now learning the BASIC language from a book entitled '30 Hour BASIC' by Clive Prigmore (recommended). You can take the 30 hour bit with a pinch of salt!

I got a great high on seeing my first program run, a little program of 4 lines:

```
10 REM DEG F TO DEG C.
20 PRINT "DEG F";TAB(15)"DEG C"
30  FOR A=1 TO 212
40  PRINT;A  ;TAB(15)(5/9)*(A-32)
50 NEXT A
```

Ah, great stuff. I then had the bit between my teeth (dentures), what would happen if in line 30 I added an STEP 0.01 ? Well, well, the numbers scrolled across the screen for what seemed like hours.

Alhough the Electron runs at 2 MHZ, it made me realize how slow 2 MHZ is if one wishes to do some "real" computing.

You Grandads, do not be intimidated by all the computer jargon. I remember picking up my first magazine and on turning the pages thinking "my God, what does all this jargon mean." Well, I can tell you that the jargon has meaning, it makes sense, bit by bit (byte by byte) it falls into place. My own personal portable computer which is protected by my thick skull is now over 70 years old, but is still running well, albeit now a little short of R.O.M. and R.A.M.

All you people who live on your own think about getting a computer. You don't HAVE to write programs - there are many games to play and many other programs to run. I play chess with my computer.

Good luck to you all.

MAILBAG

A LETTER FROM BEEBUG

The Editor of Beebug, Mike Williams, has sent a note including the following:

"I always read with interest the copies of Beeblet that we receive here in the BEEBUG office. In the July issue you printed a survey of magazines available for the BBC micro which included, I was pleased to see, BEEBUG. Unfortunately the address given is no longer correct, as a quick look at any issue over the last 12 months will quickly show.

We would appreciate it if you could print a correction, the correct address being:

BEEBUG,
P.O. Box 109,
High Wycombe,
Bucks HP10 8HQ

The subscription rate will also be £25 per annum from 1st October, 1984."

Want some free software?

I currently have available a dozen or so pieces of software for BBC "B" (provided by distributors for review purposes). Anyone who want's to get hold of something for nothing, please write to Steve Williams, 61 Disraeli Street, Epsom. All you have to do is promise to write a review for Beeblet, and you get to keep the tape!

Allister J. writes, "as the author, I refer you to two errors in the August PROC/FN library in Beeblet. The function should read as follows:

```
10 DEFFNformat(len,val)
20 =RIGHT$(STRING$(len,"0")+STR$(val)len)
```

Also, by changing the "0" to " " you have the right justification without the extra zeroes. This often looks better."

EXPAND WITH JUNK

Tony W. Napier

Got a Beeb, have you? Managed to srap up the necessary for the micro plus a cassette player and use the family TV as a VDU? Its great isn't it, but now you've had it for a few months you want to spead your wings. Alas, no more cash, not even for one little Alien Zapping game. Does despondency set in as the kids give Mum a cup of tea in bed on Fathers Day, instead of giving you the latest Micro User you can't afford yourself? (or vice versa if you are a Mum). Cheer up, read on, and learn what can be done.

I had a Beeb with 0.1 OS - couldn't spare the cash even for an upgrade to 1.2 OS - and was beseiged by bugs (VDU23;8202;0;0;0; and it was typed in correctly, Anton and Tim R.) which other people couldn't help me with because they all had 1.2 OS. However, thanks to the generosity of Paul H. of the Hawkes Bay User Group, who had a spare one, I now have 1.2 OS and the feeling of "I've got one too". This set me wondering how other goodies could be obtained. No cash! How to get cash? I had a good think and wondered what was surplus. Definitely not my wife and kids, but what about that old leather briefcase, real leather, but unused for about 22 years, and an even older leather suit case, and those wooden carpenters planes, older than I am, must be antique. And a couple of brass spring balances - saw those in an antique shop window once - and that pair of stereo headphones we don't use now that we don't have to be careful not to wake the baby. What about that pair of brass candlesticks, gathering dust, and those picture frames minus pictures, and some woodworking tools, unused since I built the garage - shall I ever build another, and if not what am I keeping them for? That mattock, sickle and pick, unused since we broke the section in, the piles of kids' games, toys and clothes, long since grown out of, and that vase (unloved and unwanted) that Aunt Mary gave us. She's gone now (poor old soul) so perhaps the vase can too, though not to the same place, but steady now, would anyone really buy this junk, and anyway it doesn't all belong to me! Days' later and $130 better off, we learn that yes, people really do buy junk. Still can't get that printer I would like, but its a start.

We (wife joins in this time) take a cold, hard look at our accumulation of possessions - looking for bigger game this time. Now those golf clubs standing in the corner of the garage since 1979, they can go, and that concrete wheelbarrow, much used when laying paths and idle ever since, and the piano sitting solidly in the family room, supporting two pot plants and a photograph of Aunt Mary (poor old soul) - a bit of pine shelving could do that job, and the kids prefer horses to Haydn. Be practical, we think, and out with those never-to-be-realised dreams, provided that you can get cash for them of course.

More days later and that printer is getting closer, and if we can do it, so can you! Put Beeblet done NOW! Get up and go to it. Good luck.

(But please drop me a line when you've tried it - Ed.)

==:

GROUND RULES FOR PROGRAMMERS

Contributed by Jill A.

1. When you don't know what you're doing, do it neatly.

2. First write your program, then draw your flowcharts.

3. Experience is directly proportional to the number of compilations plus dumps squared.

4. Documentation is useless. It indicates that you haven't been working.

5. Don't waste time looking at the hill in front of you. Climb it.

6. In case of doubt make it sound positive and convincing.

7. Do not believe in miracles; depend on them.

8.Programming is built on teamwork; it allows you to blame someone else.

No program is a complete failure;
it can always be used as a bad example.

(From the "earlier age" referred to by Brian H. ? - Ed.)

TELETEXT TITLE PAGE

Warren W.

Have you ever felt just a bit bored whilst sitting, watching, waitng while a program loads from cassette? Yes! Well it is not too difficult to create a screen full of Teletext graphics or text to gaze upon while your main program loads. This title page can be 'tacked' onto a short program which loads first and then CHAINs your main program. If RAM space is at a premium you can include things such as ENVELOPEs, VDU 23 character definitions and function key definitions for use in the main program. The first step is to design the title page. Use a sheet of graph paper or squared paper with a grid 40 wide by 25 high. Play about with a pencil and rubber until you get it looking the way you want it. Chapter 28 in the User Guide gives a brief introduction to some of the things that can be done in MODE 7. Having got your design sorted out you must then encode it. The full set of graphics shapes is given on pages 488 and 489 of the User Guide, together with the codes that generate them. Choose the appropriate control codes for the beginning of each line to set foreground and background colours, flashing, double height as required. Write down the codes for each character position, line by line, then write a program to PRINT the codes to the screen, such as in Listing #1. This program works as follows:

Line 10 sets Teletext MODE.
Lines 20 through 60 form a loop which reads each code in turn from the DATA statements, stores the code in the screen memory (which starts at address &7C00), and also stores a copy of the code in memory starting at the address &7800.
Lines 70 through 310 are the DATA statements which contain the codes, line by line, that make up the title page.

Having created a copy of the title page in memory at &7800 we have built the cart, so to speak. Now for the horse, in the form of a program that we can hitch our title page to. This program must be designed to load the title page from cassette and display it on the screen. Look now at Listing #2 which does the following:

Line 10 sets Teletext MODE.
Line 20 turns the cursor off.

Lines 30 through 50 form a loop which copies our title page, four bytes at a time, from an area of memory immediately above our BASIC program to the screen memory area starting at &7C00.

Line 60 creates and clears a text window so that we can keep tabs on how the main program is loading.

Line 70 sets control codes for alpha white at the left hand edge of the window.

Line 80 CHAINs in the main program.

Now to hitch the cart to the horse and put it in the stable. We must copy our title page down from where we parked a copy at &7800 to the area of memory immediately above our BASIC program at TOP. Then we must save our program along with the title page on tape.

Having RUN Listing #1 and then loaded or typed in Listing #2, now type:

```
FOR I%=0 TO 999 STEP 4:I%!TOP=I%!&7800:NEXT I%
```

then press RETURN. This copies the title page down from &7800 to TOP. Next type:

```
PRINT~TOP+1000
```

and press RETURN. This will give an answer, in hexadecimal, which we will call XXXX. Now get your cassette ready and type:

```
*SAVE "PROGNAME" 0E00 XXXX
```

where PROGNAME is the name that you want your main program to be loaded by, and XXXX is the result that we got above. Upon pressing RETURN your title page of 1000 bytes together with the BASIC program starting at address &0E00 will be saved onto cassette. The final thing left to do is to save a copy of your main program immediately following this on the tape. If you then rewind the cassette and type:

```
CH."PROGNAME"
```

then press RETURN, you can sit back and admire your handiwork while the main program loads.

```
10 REM TITLE2
20 MODE 7
30 VDU 23;8202;0;0;0;
40 FOR I%=0 TO 999 STEP 4
50   I%!&7C00=I%!TOP
60   NEXT I%
70 VDU 28,12,22,29,19:CLS
80 ?&7F03=135:?&7F2B=135:?&7F53=135:?&7F7B=135
90 CHAIN""
```

```
   10 MODE 7
   20 FOR byte%=0 TO 999
   30 READ char%
   40 byte%?&7C00=char%
   50 byte%?&7800=char%
   60 NEXT byte%
   70 DATA 132,157,131,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0
   80 DATA 129,157,131,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0
   90 DATA 129,157,147,154,63,47,52,63,4
7,37,63,47,37,63,47,52,53,0,0,63,47,37,4
7,63,37,62,47,52,62,47,52,63,47,37,47,63
,37,0,0,0
  100 DATA 129,157,147,154,53,0,53,53,0,
0,53,0,0,53,0,53,53,0,0,53,0,0,0,53,0,53
,0,0,53,0,53,53,0,0,0,53,0,0,0,0
  110 DATA 129,157,147,154,255,255,49,25
5,255,0,255,255,0,255,255,49,53,0,0,255,
255,0,0,53,0,111,255,52,53,0,53,255,255,
0,0,53,0,0,0,0
  120 DATA 129,157,147,154,53,0,53,53,0,
0,53,0,0,53,0,53,53,0,0,53,0,0,0,53,0,0,
0,53,53,0,53,53,0,0,0,53,0,0,0,0
  130 DATA 129,157,147,154,125,124,37,12
5,124,52,125,124,52,125,124,37,125,124,5
2,125,124,52,0,53,0,109,124,37,109,124,3
7,53,0,0,0,53,0,0,0,0
  140 DATA 129,157,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0
  150 DATA 132,157,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0
  160 DATA 132,157,0,0,0,0,0,0,0,0,0,0,1
30,112,0,114,0,101,0,115,0,101,0,110,0,1
16,0,115,0,0,0,0,0,0,0,0,0,0,0,0
  170 DATA 132,157,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0
  180 DATA 132,157,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0
  190 DATA 132,157,135,141,136,34,0,67,6
5,80,84,65,73,78,0,66,82,85,67,69,0,73,7
8,86,69,83,84,73,71,65,84,69,83,0,34,0,0
,0,0,0
  200 DATA 132,157,135,141,136,34,0,67,6
5,80,84,65,73,78,0,66,82,85,67,69,0,73,7
8,86,69,83,84,73,71,65,84,69,83,0,34,0,0
,0,0,0
  210 DATA 132,157,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0
  220 DATA 132,157,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0
  230 DATA 132,157,0,0,0,131,40,0,65,78,
0,65,68,86,69,78,84,85,82,69,0,79,78,0,8
4,72,69,0,73,67,39,83,0,41,0,0,0,0,0,0
  240 DATA 132,157,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0
  250 DATA 156,153,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0
  260 DATA 156,153,151,126,255,255,255,2
55,52,0,0,0,126,255,255,255,255,52,0,0,0
,0,126,255,255,255,255,52,0,0,151,126,25
5,255,255,255,52,0,0,0
  270 DATA 156,153,151,255,115,255,119,1
23,53,0,0,0,255,115,255,119,123,53,0,0,0
,0,255,115,255,119,123,53,0,0,151,255,11
5,255,119,123,53,0,0,0
  280 DATA 156,153,151,255,43,45,47,107,
53,0,0,0,255,43,45,47,107,53,0,0,0,0,255
,43,45,47,107,53,0,0,151,255,43,45,47,10
7,53,0,0,0
  290 DATA 156,153,151,34,47,47,47,39,0,
0,0,0,34,47,47,47,39,0,0,0,0,0,34,47,47,
47,39,0,0,0,151,34,47,47,47,39,0,0,0,0
  300 DATA 156,153,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0
  310 DATA 132,157,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0
  320 END
```

A PLUG FOR THE BEEB

Anton E.

This is not an advertising plug. That's Barsons/Whitcoulls/etc. problem. But when you look at a Beeb, it is riddled with sockets. This article discusses three of them: the power , disk and printer sockets. To wit, how to fill the aforementioned socket with plug and cable attatched thereto.

Easiest method: buy them!

The Wellington group makes up cables. It is that experience that is the basis of this article. We sell printer (data) cables for $29, disk data cables for $23 and the power cable for $12, plus $1.00 p&p. Profits (small!) of local sales go locally, but if you write into the national group, (rather than get them at our meetings) then the profit goes towards the magazine. They who drum up business shall profit therefrom.
The data cables are obtainable commercially. Wisemans (Vivian St.,Well.1) will post disk and printer data cables, but do not indulge in the time consuming occupation of making power cables. Their prices are unknown, probably about $5 more than the user group. Main centres should all have a local Wisemans equivalent.

Worst method: make them yourself!
Actually, the data cables are easy to assemble and merely require the insulation displacement plugs, the cable and a willingness to practice a vice.

DISK DATA CABLE

Parts:

(1) A 34 way insulation displacement plug (eg 3M-3414) and strain relief for same.

(2) One metre of 34 wire ribbon cable. (Any longer and you can get interference problems.

(3) 34 way edge connector. (eg 3M-3463) Two of these if two drive units are being connected.

Insulation displacement plugs are easily connected: thread the cable through the serated slot, press lightly together with fingers to get a correct fit then press firmly together in a vice. About 5 minutes to make a cable, and all you have to worry about is if it works!

PRINTER DATA CABLES

Parts:

(1) A 26 way insulation displacement plug and strain relief to match

(2) 1.5 metres of 26 way cable

(3) Printer plug: these vary and you had best look at your printer to see if it matches that drawn. Some are screwed home, but the Amphenol plug shown has notches either side for wire clips.

Both printer and disk data cables are assembled in much the same way. The numbering of plugs is: Imagine you are the socket looking at the plug. Then numbering starts top left: 1, below to 2, diagonally up to 3, below to 4, etc.
Assemble as in the diagrams. Feed the cable down into the BEEB end plug, crimp, then bend the cable down before fitting home the strain relief. The other ends may not need the cable fed in from the top. Either way should work, provided wire 1 is in its correct slot.

Both the 26 way and the 34 way plugs into the Beeb should have a strain relief clip at the back. the main use is to hold the firmly into the Beeb. The cables can be quite wasteful of wires. Firstly, only half are used: The even ones are all ground returns. But this construction is good at preventing radio interference. Then some of the disk wires are not used. 34 wires are standard, but not all are implemented on a BEEB. Pins 2,4,6,14 and 34 are not connected. All this information is interesting, but irrelevent. The printer cable is also short of wires - as you can see from the assymetric way the 26 way cable is fitted. The exta ones are not needed: one for instance is to signal back that the printer is out of paper. This is useful if the printer is miles away, but irrelevent to our needs.

POWER CABLE

The power cable is very bothersome to make up. There are 5 sources of supply: 1. AMP 6 pin plug, strain relief and 3 pins (AMP connectors, Olive Road, Auckland).

2. Self tapping screw with top filed off (for strain relief)

3. Rubber grommet (Pararubber, no.2)

4. Three core power cable (1.5 metres)

5. 4 in line plug and 3 pins (Wiseman's)

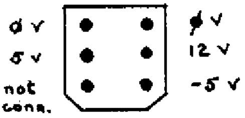
(4 core cable and two earths can be run if desired.)

Bare the wires at each end, solder seal, crimp home the pins and resolder. Push on grommet. Push pins home into AMP plug, and matching pins into plug. Fit on strain relief while cursing the grommet (not a good fit!). Test to ensure no short circuits, and all correctly fitted. Plug into Beeb. Test that the 12 volt and the 5 volt outputs are correct. THEN plug into expensive disc drive. Sit back and wonder why you bothered, instead of buying them.

The cables are made by the Wellington group, but although we can flog them off cheap to you, please feel free to make your own...

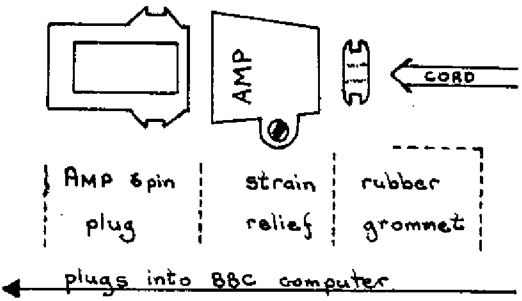
DISK DRIVE POWER PLUGS

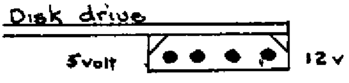
AFE'84



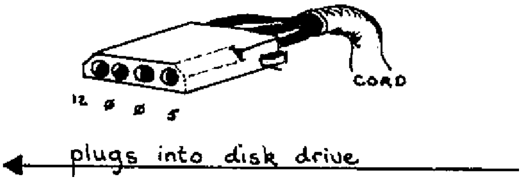
BBC Auxiliary Power Outlet

(looking under computer)



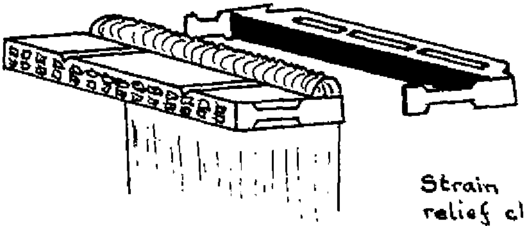


Power input



Data (ribbon) cables with insulation displacement connectors for disk drives and printers.

AFE '94

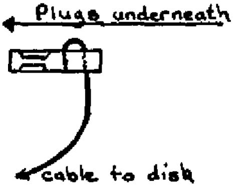


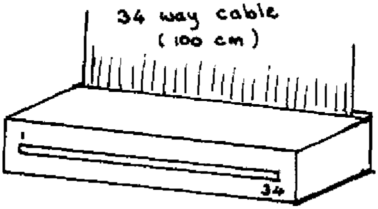
Strain relief clips on back

26 or 34 way insulation displacement connector

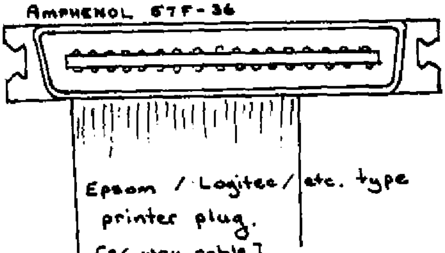
3m: 3399 - 6026
3414 - 6034

26 way for printers
34 way for disk drives





34 way edge connector
(disk drives)



Epsom / Logitec / etc. type printer plug.
[26 way cable 150 cm.]

RELOCATING VIEW

Kerry K.

One of the selling points of the 6502 Co-processor as far as I am concerned is its ability to run standard languages etc with none of the hassles of limited memory size in high resolution screen modes. For instance it would be nice to use View in mode 0 with bulk storage available. However true to form Acorn have either completely missed the point or choose to ignore it (I guess it would take someone with the source of View a whole half an hour to produce a version of HIVIEW). Presumably they will eventually catch on and we can expect an announcement of a version of View for the Co-processor (in about 12 months). In the meantime I decided to do my own. After 3 days intermittent sweating over a disassembler I had a working version of Hiview relocated to &C000. This gives approximately 48K bytes free in any mode. The program listed below allows the user to take a disk based copy of View A1.4 produced by the method described in the 6502 Co-processor user guide and produce a relocated version with the name of their choice. The program has only been tested on my own copy of View but it should work on others. If you have version A1.4 and the program complains about checksum errors, remove line 100. I make no guarantees that the resulting product is errors and I take no responsibility for loss of data resulting from its use, however it has been extensively tested and seems to function correctly.

```
 10MODE 7
 20PRINT "PROGRAM TO CREATE A HIGH VERSION OF VIEW"
 30HIMEM=&3C00
 40INPUT "ENTER INPUT FILE";I$
 50OSCLI "LOAD "+I$+" 3C00"
 55CS%=0
 60FOR I%=0 TO &2C12
 70CS%=(CS%+I%?&3C00) AND &FFFF
 80NEXT I%
 90PRINT"CHECKSUM IS ";~CS%;
100IF CS%<>&80A THEN PRINT" SHOULD BE 80A":STOP ELSE PRINT
110INPUT "ENTER OUTPUT FILE NAME";V$
120FOR I%=1 TO 714
130READ A%
140A%?&3C00=A%?&3C00+&40
150NEXT I%
160OSCLI "SAVE "+V$+" 3C00 +2C12 C000 C000"
```

```
170PRINT"COMPLETED."
1000DATA &2,&5,&56,&6F,&72,&77,&9E,&A4,&AC,&117
 1010DATA &120,&141,&144,&14D,&150,&15A,&15D,&165,&168,&16B
 1020DATA &187,&18D,&196,&1AC,&1B4,&1BE,&1D8,&1F8,&207,&223
 1030DATA &256,&25C,&282,&285,&28A,&28D,&290,&293,&296,&2AB
 1040DATA &2AE,&2BA,&2CD,&2D8,&2DD,&2E0,&2EB,&2EE,&2F1,&2F4
 1050DATA &2F9,&2FC,&2FE,&305,&308,&30B,&30E,&311,&316,&319
 1060DATA &31C,&31F,&328,&32B,&332,&335,&338,&33B,&345,&350
 1070DATA &353,&356,&35D,&361,&364,&367,&36A,&36F,&376,&379
 1080DATA &37C,&381,&384,&387,&39C,&3B4,&3EE,&3F6,&3F9,&3FE
 1090DATA &401,&404,&407,&40C,&40F,&421,&428,&42B,&42E,&433
 1100DATA &439,&43C,&441,&454,&45B,&461,&464,&46B,&480,&483
 1110DATA &48E,&491,&494,&497,&49A,&49D,&4B2,&4BF,&4C2,&4C5
 1120DATA &4D0,&4D7,&4DC,&4E2,&4E5,&4FD,&500,&503,&506,&509
 1130DATA &522,&526,&53F,&557,&55C,&55F,&562,&565,&57C,&5BD
 1140DATA &5C0,&5D6,&5DB,&5E7,&5F8,&603,&62B,&63D,&642,&645
 1150DATA &648,&651,&654,&659,&664,&684,&689,&6BB,&6C4,&6C7
 1160DATA &6D1,&6DB,&6E3,&6EA,&6ED,&6FD,&700,&709,&70C,&711
 1170DATA &717,&71A,&741,&760,&773,&776,&77F,&7F3,&7F6,&7FF
 1180DATA &808,&812,&817,&833,&83A,&84B,&86E,&884,&891,&8A7
 1190DATA &8C3,&8C6,&8D2,&8E5,&8FF,&923,&955,&9C4,&A4C,&A55
 1200DATA &A8F,&A9A,&A9D,&AA0,&AAD,&AB0,&B24,&B27,&B3C,&B58
 1210DATA &B5D,&B99,&B9C,&BCE,&BDB,&BE0,&BE3,&BE6,&BE9,&C17
 1220DATA &C1A,&C23,&C29,&C33,&C3A,&C49,&C5D,&C60,&C63,&C68
 1230DATA &CA4,&CC0,&CC3,&CCA,&CD7,&CDA,&CFA,&D05,&D08,&D0B
 1240DATA &D11,&D14,&D17,&D1A,&D34,&D8C,&DA8,&DBD,&DD6,&DE7
 1250DATA &DEA,&DFF,&E08,&E0B,&E14,&E20,&E39,&E96,&EAA,&EBD
 1260DATA &ED8,&EE0,&EEB,&EEE,&EF3,&F0D,&F10,&F20,&F23,&F29
 1270DATA &F3A,&F3F,&F44,&F4F,&F52,&F57,&F5A,&F6D,&F74,&F77
 1280DATA &F7C,&F7F,&F82,&F85,&F88,&FA3,&FA6,&FA9,&FAC,&FB1
 1290DATA &FB4,&FC8,&FD4,&FDE,&FE3,&FE6,&FE9,&FEC,&FEF,&1023
 1300DATA &1026,&103A,&1047,&1054,&109A,&109D,&10A0,&10A3,&10A6,&10BA
 1310DATA &10D9,&110F,&114D,&115B,&115E,&1168,&1171,&117E,&118D,&1195
 1320DATA &119A,&11A2,&11AA,&11B2,&11BA,&11C2,&11CA,&11D2,&11EC,&11F3
 1330DATA &11F6,&11FB,&1203,&123C,&124F,&125D,&128A,&1297,&12B3,&12BA
 1340DATA &12C2,&12CF,&12DD,&12E6,&12EC,&12EF,&12F2,&1311,&1329,&135D
 1350DATA &136F,&137D,&13A2,&13F1,&145F,&14A2,&14AF,&14C9,&14E1,&1554
 1360DATA &1557,&155A,&155D,&1562,&1580,&1583,&1586,&158D,&15A6,&15A9
 1370DATA &15AC,&15AF,&15C7,&15DC,&15E7,&15FA,&1608,&1628,&1642,&1645
 1380DATA &1664,&1669,&166E,&1671,&169A,&16D3,&16D6,&16D9,&16DC,&16E3
 1390DATA &16EA,&1705,&170C,&171C,&1723,&1732,&1740,&1747,&1757,&176C
 1400DATA &177F,&1788,&1793,&17A4,&17B3,&17B6,&17D8,&17DB,&17EF,&17F2
 1410DATA &17FF,&180A,&1812,&1818,&1829,&1846,&184D,&1850,&185B,&185E
 1420DATA &1861,&1878,&187B,&18AC,&18D3,&18E7,&1900,&1903,&190C,&1915
1430DATA &191C,&192C,&1931,&1934,&1943,&194E,&1959,&1966,&196D,&1973
 1440DATA &1976,&1981,&1984,&1987,&198A,&199B,&199E,&19A1,&19A4,&19A7
 1450DATA &19AE,&19B1,&19B4,&19BF,&19CA,&19EA,&19ED,&19F0,&19FB,&1A02
 1460DATA &1A27,&1A38,&1A3F,&1A42,&1A49,&1A4C,&1A4F,&1A52,&1A5C,&1A7B
 1470DATA &1A86,&1A93,&1A96,&1ABC,&1ABF,&1AC8,&1ACB,&1ACE,&1AD1,&1AD4
 1480DATA &1AD7,&1ADA,&1ADF,&1AE8,&1AEB,&1AF2,&1AF5,&1AF8,&1AFB,&1B00
 1490DATA &1B07,&1B0A,&1B2B,&1B43,&1B8C,&1B93,&1B9F,&1BA2,&1BA8,&1BAE
 1500DATA &1BB1,&1BC6,&1BE3,&1C09,&1C34,&1C48,&1C4B,&1C74,&1C85,&1C8C
 1510DATA &1C8F,&1C92,&1CBC,&1CDE,&1CED,&1CF4,&1D0E,&1D11,&1D14,&1D17
 1520DATA &1D26,&1D2D,&1D46,&1D49,&1D61,&1D64,&1D92,&1DB8,&1DCF,&1DD2
 1530DATA &1DDB,&1DDE,&1DE1,&1DE4,&1DE7,&1DF3,&1E24,&1E31,&1E44,&1E7C
 1540DATA &1E85,&1EC5,&1F12,&1F22,&1F45,&1F4C,&1F88,&1F8B,&1FB2,&1FBB
 1550DATA &1FC2,&1FC9,&1FD8,&204E,&206C,&2071,&2077,&207A,&20D9,&20E4
 1560DATA &20FD,&210C,&212D,&2133,&213A,&215D,&216D,&2170,&217C,&2187
 1570DATA &21AE,&22AB,&22AE,&22B1,&22BE,&22C1,&22D6,&22FB,&233F,&2344
 1580DATA &2385,&238A,&23A6,&23C3,&23FE,&2415,&241C,&243A,&245E,&2463
 1590DATA &24C2,&24C5,&24C8,&24CB,&24D5,&24D8,&2514,&2525,&2536,&2539
 1600DATA &2541,&257B,&25C7,&260A,&265A,&2674,&267D,&2689,&26AC,&26B8
 1610DATA &26C1,&26CE,&26EA,&26EF,&2738,&274B,&274E,&2785,&279C,&2826
 1620DATA &2865,&2867,&2869,&286B,&286D,&286F,&2871,&2873,&2875,&2877
 1630DATA &2879,&287B,&287D,&287F,&2881,&2883,&2885,&2887,&2889,&288B
 1640DATA &288D,&288F,&2891,&2893,&2895,&2897,&2899,&289B,&289D,&289F
 1650DATA &28A1,&28A3,&28A5,&28A7,&28A9,&28AB,&28AD,&28AF,&28B1,&28B3
 1660DATA &28B5,&28B7,&28B9,&28BB,&28BD,&28BF,&28C1,&28C3,&28C5,&28C7
 1670DATA &28C9,&28CB,&28CD,&28CF,&28D1,&28D3,&28D5,&28D7,&28D9,&28DB
 1680DATA &28DD,&28DF,&28E1,&28E3,&28E5,&28E7,&28E9,&28EB,&28ED,&28EF
 1690DATA &28F1,&28F3,&28F5,&29E9,&29EB,&29ED,&29EF,&29F1,&29F3,&29F5
 1700DATA &29F7,&29F9,&29FB,&29FD,&29FF,&2A01,&2A03,&2A05,&2A07,&2A09
 1710DATA &2A0B,&2A0D,&2A0F,&2A11
```

EDIT YOUR OWN USER DEFINED CHARACTERS

Neville T. Christchurch

How much easier would it be if you could edit your user-defined characters directly on the screen instead of having to redefine the whole thing!

This short routine not only allows you to enter and edit your characters by drawing them as they look on the screen, but it is also short enough to be incorporated as a single key definition!
First of all, lets have a look at how a user defined character is stored in memory.

The BBC micro uses one page of memory from &C00 to &CFF to store the information. This is "divided" into sections of 8 bytes, each byte corresponding to one row of a character. The first 8 bytes relate to CHR$225 etc... right up to CHR$255 (allowing you to define 32 characters).

OK, heres the program you can use to define a character:

```
 10 MODE 7:INPUT TAB(10,10);"Enter which character ",char
 20 X=0:Y=0:MODE 7
 30 REPEAT
 40 PRINT TAB(X,Y);
 50 Q=GET
 60 X=(X+(Q=90)-(Q=88))MOD8
 70 Y=(Y+(Q=58)-(Q=47))MOD8
 80 X=X-(X<0)*8:Y=Y-(Y<0)*8
 90 ?(HIMEM+X+Y*40)=Q=32
100 UNTIL Q=83
110 I=0
120 FOR Y=0 TO 7
130 C=0
140 FOR X=0 TO 7
150 C=C+(2Δ(7-X) AND ?(HIMEM+X+Y*40)=255)
160 NEXT X
170 PRINT TAB(27,Y);C
180 ?((char-224)*8+&C00+I)=C
190 I=I+1
200 NEXT Y
```

Of course, this could be compressed into one key definition by using abbreviations, as follows:

```
*K.0X=0:Y=0:VDU22,7:REP.:P.TAB(X,Y);:Q=GET:X=(X+(Q=90)-(Q=88))MOD8:
   Y=(Y+(Q=58)-(Q=47))MOD8:X=X-(X<0)*8:Y=Y-(Y<0)*8:?(H.+X+Y*40)=Q=32:
   U.Q=83:I=0:F.Y=0TO7:C=0:F.X=0TO7:C=C+(2Δ(7-X)A.?(H.+X+Y*40)=255):
   N.:P.TAB(27,Y);C:?((g-224)*8+&C00+I)=C:I=I+1:N.‖M
```

__NOTE:__ If you are using the key definition, you will have to define the variable 'g' as the character you want to define beforehand eg. type "g=224" then press the function key to define CHR$224.

Now, lets have a look at how it works.

Looking at the program, first of all it asks you which character you want to define, and stores this as the variable 'char'. The program works by showing you a blank screen which you can move around using the keys:

Z	(left)	X	(right)	S	(finished)
:	(up)	/	(down)	<space>	(print a block)

Thus, you move around an 8X8 grid, pressing the space bar to insert a block and in this way draw the character you want to define. To wipe out a block, just move over the top of it. When you have finished, press the 'S' key, and the character will be defined and stored in memory as CHR$char.

X and Y are the variables where your cursor is present on the grid, so always hold a value between 0 and 7. The program will loop in the REPEAT...UNTIL in lines 30-100 until the 'S' key is pressed (ASCII 83 - line 100).

In this loop, your cursor is printed (line 40), then the BBC waits for you to press a key (line 50), storing the ASCII value in the variable Q. The new X and Y, coordinates are worked out in lines 60-80. For those of you who are not familiar with the method used, it works as follows:

The ASCII value for Z is 90, X is 88, : is 58 and / is 47.

So, if Z is pressed (left), (Q=90) will be true, returning a value of -1, but (Q=88) will be false, returning a value of MOD 0. Line 60 then becomes X=X+(-1)-0) MOD 8, or X=X+(-1) 8. Hence, X will decrease by 1, a movement left. The MOD 8 just ensures that you stay on an 8X8 grid.

Or, if the X is pressed (right), (Q=90) is false (0), but (Q=88) is true (-1). Line 60 is now X=X+0-(-1)) MOD 8 or X+(X+1) MOD 8. Hence, X will increase by 1, a movement right.

The same applies for up and down movement, in line 70.

Line 80 is a "wraparound" feature, ie if you go off the left edge of the grid, you will appear on the right edge, or if you go off the top of the grid, you will appear on the bottom.

It works like this:
*If you are off to the left side, X is -1, so (X<0) is true (-1). Line 80 will then make X=X-(-1)*8, or X=X-(-8) or X=X+8. Since X is -1, the new value for X is -1+8, or 7 (the right hand edge).*

Line 90 comes into effect when you press the space bar (Q-32), as the ASCII of a space character is 32. It works by poking a value directly into screen memory.

The start of screen memory is held in the value of the pseudo-variable HIMEM, so, when the space bar is pressed, we want to display a block (CHR$255 in MODE 7, but if space is not pressed we want to display a blank. HIMEM points to the top left hand corner of the screen - see figure 1. This is made easy for us, because in mode 7, each character is stored directly in screen memory in its ASCII form.

To figure out which byte of memory our cursor is currently at, we use HIMEM (the starting value)+X (how far from the left side of the screen we are) + Y*40 (Mode 7 has 40 characters per line, and Y holds the number of lines down the screen that we are).

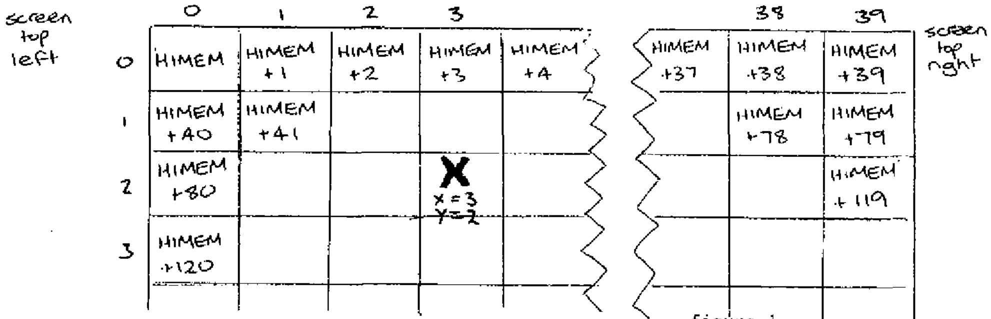


figure 1.

Look at the above diagram. HIMEM is the top left screen corner. As there are 40 characters per line, and 1 byte per character in Mode 7, the Mode 7 screen top right is at HIMEM+39, and so the beginning of the second line is at HIMEM+40. The numbers top and down the left hand side are the X and Y co-ordinates of the screen. So, by way of example, when your cursor is at position 'X' on figure 1, the values of X and Y will be 3 and 2 respectively. Using our formula, the character currently printed here will be at HIMEM+X+Y*40, or at HIMEM+3+80, or at HIMEM+83.

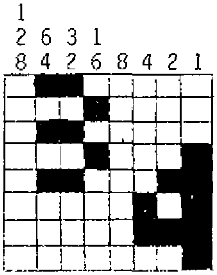
We can print a character anywhere on the screen by poking its ASCII value into the corresponding memory location. So, when the spacebar is pressed, we want to print a block (CHR$255), so we poke 255 into memory. Hence, if the spacebar is pressed, (Q=32) in line 90 is true (-1), and the memory location has -1 written to it. As this memory location can hold only one byte, writing -1 to it stores &FF (or 255) there - check this by PRINT ~-1 (print the value in hex of -1).

Since CHR$255 is a solid block in Mode 7, and 255 has been stored in this location, a white block appears at the cursor when the spacebar is pressed. If the spacebar is not pressed, line 90 stores a 0 to that location (because (Q=32) will be false or 0), and so the position at the cursor will hold a blank (now you see why your solid white block disappears when you go over it again with the cursor!).

OK, that takes care of how you draw your character on the screen. Now comes the REALLY tricky bit - storing what you have drawn as a character in the memory. This is taken care of by the rest of the program. To store the character, we must calculate a value for each row, just like you have to do when working out a definition on paper. (Editor's note : actually, its easier to read each group of four cells off directly as a hex number - with a little practice you soon remember the patterns...). These values are then stored in the &C00-&CFF area of memory to define the character.

So we need 2 loops - one to read across the character and one to read down (lines 120 and 140). I is a variable telling us which row we are up to, and holds a value between 0 and 7. C is calculated as the value of that row.

I am assuming that you are familiar with how you find the values for a character by adding across the row, with the rightmost bit having a value of 1, and this doubling as you move left until the leftmost bit has a value of 128. See figure 2 for clarification.



value=64+32 =96
value=16 =16
value=64+32 =96
value=16+1 =17
value=64+32+2+1 =99
value=4+1 =5
value=4+2+1 =7
value=1 =1

Define using:

VDU 23,x,96,16,96,17,
99,5,7,1

figure 2.

Line 150 calculates the value of each row. The column value is calculated by 2^(7-X) ie when you are on the first column, X=0, so the value is 2^7, or 128. When you are in the right most column , X=7, so the value is 2^0, or 1. By ANDing the value of the column with whether there is a block at this position or not, and adding these, we end up with the correct value for that row.

If there is a block at that position, ?(HIMEM+X+Y*40)=255 will be true, and return the value of -1. If, say, the column is 64, 64 AND 0 gives 0, so C will have 0 added to it. See how it works ? Any number

AND -1 gives that number back, and any number AND 0 gives 0, so we end up with the correct total for that row!

Line 70 prints the row value on the screen, and then line 180 pokes the value into user-defined character memory before increasing I and moving on to the next row via the loops. BUT, how do we know where to store the value of the row to define our character?

Here comes another interesting piece of maths:

As mentioned before, we must store the value somewhere in page &C00, so we can use &C00 as an absolute starting value. Look at line 180. Remember, the variable char holds the character we are defining. So if char is 224 (the first character in page &C00), by subtracting 224 we get 0, which points to the first byte in page &C00. Or, if we were defining CHR$225, we would want to be concerned with the second set of 8 bytes in page &C00, so by subtracting 224 we get 1. Now if we multiply the (char - 224) by 8, and add this to &C00, we get the starting memory position of the character we are defining.

For example, to define CHR$255, we would be looking at the last 8 bytes of page &C00, or &CF8-&CFF. We can find this position by using (char - 224)*8+&C00, as this gives us (255-224)*8+&C00, or &C00+248. As 248 in hex is &F8, &C00+&F8 gives us &CF8, the correct starting position.

So, we must poke the values of the 8 rows of our character into the 8 consecutive bytes starting at the value given to us by (char - 224)*8+&C00 in order to automatically define our character.

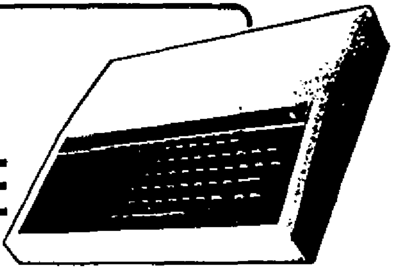
This is done using our row counter, I. For example, the value of the 5th row of the character in figure 2 is 99. Because this is the fifth row, we want to place it in the fifth byte from the starting position. When we are calculating this value, I has the value 4, so by adding I to our little formula, we can get to the fifth byte.

So, the final formula is (char-224)*8+&C00+I, which is what appears in line 180. C is the value calculated for that row in line 150.

Well, that's how you manipulate data for a user defined character. In the next article I will describe another short program, but this one will draw an already-defined character on the screen, and allow you to edit it and restore it in memory. Quite a time saver!

Molymerx

BBC SOFTWARE



The MOLYMERX COLOUR SOFTWARE CATALOGUE is ready !

Pages of descriptions of new powerful Application and Utility software for $3.00

and THAT's NOT ALL -

as we publish new software we send you free UPDATES. Our offer is to keep you abreast of NEW products, NEW services and our regular special prices.

Some of the products in the latest edition of our catalogue !

DISK FILE

True Random Access File DATABASE

WAR GAMES

Napoleon Crusaders Emperor

SHUTTLE II

DISK ZAP

MONITOR

Fly COLUMBIA II from space to a Desert landing (if you can !)

Fill in the coupon and return to:

Molymerx

Molymerx Ltd.
P.O. Box 60-152
Titirangi,
AUCKLAND
NEW ZEALAND

(817) 4372

Please send ☐ copies of the Molymerx Software Catalogue.

Please use BLOCK CAPITALS

Name/Company ....

Address ....

..................Zone..................

I enclose $..................@ $4.50 per copy. TRS-80 Catalogue

I enclose $..................@ $3.00 per copy Colour Catalogue

Australasian distritors for: Aardvark, Acorn, Big Five, C.A.U., A.J. Harding, Hexagon, ICR Futuresoft, Logical Systems, Micro Systems Software, Mumford Micro Systems, Pioneer Software, Understanding, ABC, Powersoft.

Dealer Enquiries Welcome

STRUCTURE DIAGRAMS

(Again)

Bryan H.

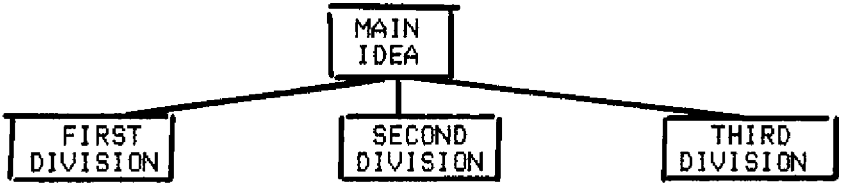
After two articles about structured program design you may well ask: why are all these people raving about it? The reason is that structure diagrams are an aid to thought (and programming), not a hinderance. They should not be an expert's tool - everyoe should be using them.

Most books published today and some courses use the older flow charts if they use anything. Flow charts are the relics of an earlier age. The style of programming they were invented for has now been superseded by advances in computer science, and in knowledge of human thought processes. B.B.C BASIC, along with ALGOL, Pascal, LOGO etc have the facilities to use structured programming methods directly. The BBC "Proc" is the ideal tool implement structured programming.

The method of using the boxes for programs is described in detail in the May 1984 issue of Beeblet. By using the shapes representing Action, Repetition and Decision any program or idea can be implemented.

The process is one of successively breaking down the main task into sub-tasks.

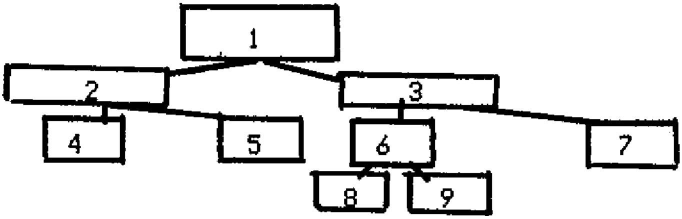
eg.



The sub-tasks are then successively broken down until it is not productive to break them down any further. This way of working gives the process its name "Top Down Programming". (Now you understand what its about you can use both terms to impress non programmers.)

Once your diagram is complete it is convenient to write your program in the order it is executed. For the simple diagram below, working from the top you should have broken it down in the order

1, 2, 3, 4, 5, 6, 7, 8, 9. Your program will follow the order 1, 2, 4, 5, 3, 6, 8, 9, 7.



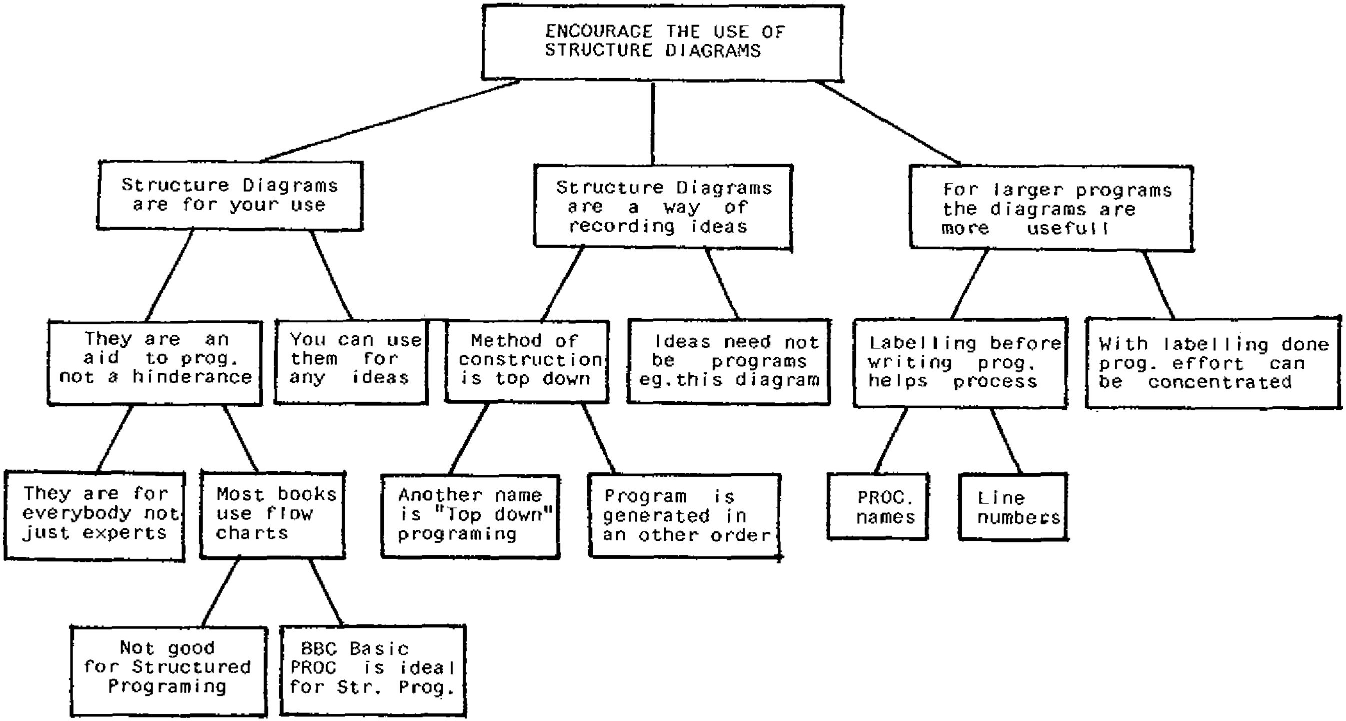
To demonstrate that structure diagrams can be used anywhere, here is the diagram I drew before starting this article. There has been some expansion and reorganisation along the way but I hope that human readers are a bit more adaptable than computers.

As programs become longer the diagrams become more useful. It is true that small programs can be written without formal planning or structuring. Medium and large programs are much easier to produce if the planning is done first. Once the structure diagram is produced and the coding starts the programmer only needs to concentrate on one small area at a time, literally filling out the coding for one box before going on to the next.

The diagram can be used in two more ways to speed up the coding process. Each box can have a line number assigned and procedure names can be invented and written on the appropriate (square) boxes - See Warren W.'s article in the June 1984 Beeblet for examples of procedure names inside the boxes. I put the line numbers outside the boxes, on the top of each box, and use the full range of line numbers. eg. major divisions would start at 10,000 20,000 30,000 40,000.

If you start off this way you can give each box a line number without fear of clashes. The only disadvantage is the loss of a few bytes of memory and perhaps a few square mm of printing space.

Structure diagrams are also a useful way of communicating program ideas. Maybe you you are only writing for your own machine. Other BBC owners may be interested in your program, however users of other machines (friends, your local school...) will want to know how it works - they can do their own coding and use sections of "standard" BASIC from the listing.


ENCOURAGE THE USE OF STRUCTURE DIAGRAMS
Structure Diagrams are for your use
Structure Diagrams are a way of recording ideas
For larger programs the diagrams are more usefull
They are an aid to prog. not a hinderance
You can use them for any ideas
Method of construction is top down
Ideas need not be programs eg.this diagram
Labelling before writing prog. helps process
With labelling done prog. effort can be concentrated
They are for everybody not just experts
Most books use flow charts
Another name is "Top down" programing
Program is generated in an other order
PROC. names
Line numbers
Not good for Structured Programing
BBC Basic PROC is ideal for Str. Prog.

FILES. PART THREE: RANDOM FILES.

Steven R.

True or false?

1) The BBC has two different types of file: sequential and random.
2) The BBC only processes and stores one type of file: sequential.

Answer: both are true. Confused? Let me explain.

In my last article, I told you that the difference between a sequential file and a random file is that with a sequential file you can't go back and reread or rewrite something, and with a random file you can. If you remember, I also said that I was going to bend the truth a little. The truth is, sequential files and random files are the same physically, but different logically. In other words, the BBC stores them in the same way, but you treat them differently. If you compare what the BBC has to do to print something physically, taking care of scrolling, calculating screen positions etc, compared to what you think of it as doing logically, printing at TAB(3,4) on the screen, you will get an idea of what I mean.

The difference lies in how you use the file. To use any file, you say OPENIN or OPENOUT, and then start to read from or write to it as appropriate. Usually, you just read everything in the order that it was written. But if you OPENIN any file on disc, you can go back and re-read an item, or change an item. In fact, you can read or write an item anywhere in the file, jumping all over the place at random if you want to, so long as the file is on a disc, and so long as you OPENIN'ed the file. This is called random access.

You can do this by changing PTR#, a special pointer that BASIC uses to keep track of where you are in the file. PTR# can be looked at by saying "position% = PTR#channel%", or changed by saying "PTR#channel% = position%".

There is a slight complication however. An item in a file usually takes several bytes to store. This means that you must know exactly where an item starts before you can read something that is not rubbish, or write something without corrupting the file. Think about it. How do you jump over 100 strings (of different sizes) and numbers in a file, and stop at the exact right place? It is just a little bit tricky, isn't it? If there are only integer numbers, then it's easy, because integers are always stored in the same number of bytes. The same applies to real numbers. Strings, on the other hand, are usually of varying lengths, and you'd need ESP to know the length of 100 strings that you haven't looked at.

How do you tell where the pointer must be placed? One way is to store your information in a set order. This way, you can predict where the start of any item will be by multiplying the number of items that you are skipping by their length, and adding your current position. You will have to know how many of each type there are, and how long each string is. You can then calculate where to set PTR#.

Another way is to store pointers in the file. Each pointer points to the start of an item or group of items in the file, so to jump somewhere, you just set PTR# to whatever the stored pointer is. This could mean following a chain of pointers through the file until you get to the place you want to be. If you want to go backwards, it may also mean that you need a pointer to go backwards, which will use up more room. Both of these will depend on how you designed the file. The biggest drawbacks with pointers are first, if you move something in the file, you have to change all the pointers that point to it so that you can find it again. Secondly, pointers use space in the file. The advantage with pointers is that the order of the information doesn't matter. You can insert something in the middle of the file by putting it at the end, and altering the pointers to point to it. You can delete something just as easily by changing the pointers to miss out what you want to delete. This is called logical deleting, because the item is still physically there. Logically deleted items can be undeleted again so long as you know where they are.

The other way of finding information in a file is by having a separate index file, which contains a key, and a pointer to the data file, for each item. The key can be anything that is unique in the file, like a telephone number, a surname (not as good), or an invoice number. To find something, you search the index file for the correct key, and then use the pointer with that key to set PTR# in the data file. The index file should have keys and pointers of a fixed length, and should be in some sort of order, so that you can use one of the fast searches to find the key. Searching techniques may be the topic of another article, depending on the Editors' persuasiveness.

There is one last method. You can design your own way of storing things. If you are only storing one type of information, or if the order of variables will always be the same, then you don't need the variable type indicator that BASIC stores. If you are only using numbers of 255 or less, then you can store them in single bytes. If you are storing a lot of text, you might want to compress it to save room, and therefore to store it in your own way. The drawback is that you have to handle the storing and deciphering of the information. The advantage is that you are saving room, and possibly speeding things up, or making the logic of a program easier.

Before I talk about the uses of random access files, I must warn you of some pitfalls. First, if you are overwriting something, make sure that what you write is the same length as what was already there. If it isn't, you will probably corrupt the file, which is almost as bad as deleting it. Second, make sure that PTR# is set to the exact byte. If it isn't, then you will be reading garbage, or writing in the wrong place, which will corrupt the file. Third, the token for OPENIN that BASIC I uses is different for BASIC II, so that if you run a program on a BASIC I machine, it may not work properly on a BASIC II machine. Also, BASIC II uses OPENUP for random access, NOT OPENIN. Fourth, when you open a new file to write to it, BASIC allots a set amount of space on the disc. It is quite easy to try to write past the end of that area later, using random access, which will give you a "can't extend" error. This will only happen if your file is not stored last on the disc. If it is, then BASIC will just give you some more room.

What do you use a random access file for? Well, anything that can be done with a sequential file can be done with a random file, because they are really the same. Random files come into their own when you want to do something fast or tricky. The standard example is something like:

> Given a phone number, find the name, when the file is in name order.

With a seqential file you would read all the way through the file until you finally found the correct phone number, and then get the name that went with it. With a random file, assuming that you had an index file of phone numbers, you would find the required number in the index file, using a fast search like a binary chop (...the imagination boggles...Ed.), and then go straight to the right place in the main datafile, using the pointer stored in the index file. Often, because you are only storing the key to each block of information rather than the whole lot, you can load the index file into memory. This often speeds up the search and the reading of your information by an incredible amount.

For those of you who are a little more adventurous, try this. Create a file containing data on all of your records and tapes, organised in such a way that you can:

1. Read the titles of all records that you have recorded on tape

2. Read the titles of all tapes and records with a given performer on them

3. Read the titles of all records or tapes produced by a given performer

Easy? Well, I haven't told you the last requirement. You must only read the record/tape information for the category that you were looking for. You can't read information about other records or tapes, and then just print the ones that you want.

Harder, isn't it? It can be done using pointers for each category of information, that point to the next item in that category. Some record/tape information will have several pointers to or from it, one for each category. You will find this difficult to set up, but fast to use once it has been set up.

Well, that's all, folks. (It isn't, of course... Ed.). Remember that random files exist when you are designing things and remember the usefulness of pointers, whether in a sequential or random file.

Comment:
Could someone please explain these "pointers" : exactly what are they and how do you go about using them ? -Ed.

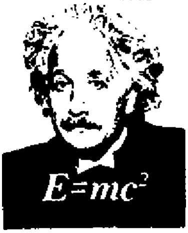
BBC PERIPHERALS AND SOFTWARE
- AROUND NEW ZEALAND

acorn electron

CHECK EINSTEIN'S EXTENSIVE RANGE OF PRINTERS AND OTHER EQUIPMENT FOR YOUR BBC OR ELECTRON

*177 Willis Street, **Wellington**
Phone Raju, Michael or Malcolm 851055.
*154 Broadway Ave, **Palmerston North**
Phone Richard 64108.
*Cnr King & Egmont Sts, **New Plymouth**
Phone Bill or Sally 85528.
*Shop 41, First Floor, Cashfields Mall, **Christchurch**
Phone Mike 66442.
*inside DIC Ltd, Garden Place, **Hamilton**
Phone Albert 81969.

Einstein
SCIENTIFIC LTD



The computer world made simple.

HELP

Michael G. of Feilding writes: "One problem that I have had when loading games from tape to disc one or two have come up with BAD MODE line.... If I alter the mode the colour of course changes. Could one of the members help me?"

Michael, I think the problem relates to the extra memory which the disc drive grabs compared to the tape system. For some games, typing in something like Page=&1200 before loading them works. Alternatively, once you have loaded them from disk you can type in *T. (switching back to tape operation, and memory allocation) and then move the program down in memory. One game I have includes the following function key definition to move the program down:

```
    1  *KEY0 *TAPE|MFORI%=0TO TOP-PAGE STEP4:I%!&E00=I%!&190
 0:NEXT|MPAGE=&E00|MEND|MLOMEM=TOP|MGOTO10|M
    2  *FX 138,0,128
    3  ONERRORDELETE 1,4
    4  ERROR
```

Andrew Ammundsen of 5 Joanna Place, Hamilton, writes, "Having bought a number of game programs and then getting tired of playing them, would it be possible to sell or swap them with other BBC owners. I have such games as Cyclon Attack, Vortex, Twin Kingdom Valley, Lunar Rescue, Planetoid, Hunchback and Danger UXB. I am not sure if you agree with members swapping or selling software through your magazine, but after spending lots of money on games it seems a waste that they now get little use."

The main problem here Andrew is the temptation to retain a copy when passing on the original game. This would be a clear breach of copyright which Beeblet could not condone. However, if people wishing to swap games were to provide an undertaking that they would not keep a copy, I think this should be OK. I suspect you may get into a slightly more dicey situation in the case of resale - perhaps one of the dealers might like to comment.

Alan J. of Oamaru sent the following "Good news down south, we

(Waitaki Boys High School) have just put in an order for a 15 station Econet and I hear that Southland Boys, Girls and James Hargest schools are doing something similar in Invercargill. Add to that the 15 just installed at Christchurch Girls and there is a rapidly growing user group which could soon ease the burden of lack of news for the magazine.

....have you come across problems with discs formatted on different machines - trying to read them on your own? I had happily assumed that the only differences were 40/80 track and felt utterly confident with a switchable disc drive of being able to read anything on a 5.25" floppy (apart from network format, of course). Not so simple however - I observed a colleague at a course having troubles with different machines from different schools and my neighbour's 40 track (!) won't pick up through my system (although when I format a 40 track for him, he can run my discs!). Is there any way of identifying a "friendly" DOS without waiting for the drive's groans and "disc fault at...." etc? ".

Alan, I have passed your note to Tony K. of Barsons (and Chairman of the Auckland branch) who, in the June issue, offered to run a "Tell Tony" column. Just a reminder: You are all welcome to send your problems, even if you suspect that they are trivial, to -

Tell Tony
MICROLOGIC
P.O. Box 580
Manurewa.

I have not encountered such problems on local BBCs equipped with 80 track discs and Acorn DFS. I know however that a common problem occurs when the disc drive heads get out of alignment (the same happens on cassette recorders). The drive can read disks it has written to and often other people's disks, but no-one else can read its discs. Hopefully Tony will be able to shed further light on this one!

==

Lyndon McEntee of Colac Bay School, Riverton, Southland has forwarded the following request:

" I have nearly completed an index to all BBC articles, reviews and news items (ie. everything except the ads) in Micro User, Acorn User, Beebug and Electronics & Computing Monthly. However, I have been unable to obtain any 1982, July 1983, or August 1983 issues of Acorn User. Anyone who can lend me any of these for one week will receive a free disk copy of my completed file."

```
FGFGFGFGFGFGFGFGFGFGFGFGFGFGFGFGFGFGFGFGFGFGFGF
FG                                           FG
FG   FG Software PO Box 6662 Auckland         FG
FG                                           FG
FG   Don't pay $40.00 for software!           FG
FG                                           FG
FG   We have in stock the new Electron        FG
FG   compatible version of the famous         FG
FG   DOCTOR SOFT 747 Flight simulator         FG
FG   (consistently number 1 game in UK)       FG
FG        ONLY $24.95!!!! ($32.95 on disk)    FG
FG                                           FG
FG   Look out for our products in Gilbert     FG
FG   Electronics and The Computer             FG
FG   Terminal (opening soon in Birkenhead)    FG
FG                                           FG
FG   We are also agents for Barson Computers FG
FG   (i.e. Acorn,Micro Power etc)             FG
FG                                           FG
FG                                           FG
FG      SEND SAE FOR OUR FREE CATALOGUE       FG
FGFGFGFGFGFGFGFGFGFGFGFGFGFGFGFGFGFGFGFGFGFGFGF
```

HALL OF FAME

PUBLISHER:	GAME	HISCORE	SCORER
Acorn	Aviator	150	Kendall
	Countdown	227	Tim R.
	Hopper	27,196	Jenny B.
	Meteors	24,090	Duncan's flatmate
	Monsters	93,560	Linsay F.
	Starship Command	7,073	Robert N.
A&F	Cylon Attack	68,340	Robert N.
Badger	Lunar Lander	245,967*	Duncan's flatmate
Level 9	Colossal Cave	687	Steve W.
	Dungeon Adv.	330	Anne et al
Pace	Fortress	20,600	Kendall
Prog.Pow	DangerUXB(Lvl 1)	286,690	Steve W.
	Felix the Factory	7,000	Dianne S.
Sup.Soft	Alien Dropout	24,200	David E.
	Q*BERT	30,260	Dianne S.
	Roadrunner	70,600	Lois S.

*Hardest level.

HALL OF FAME

CLASSIFIED AD

WANTED Any record program to keep church membership records (name, address, phone number and other church related information). Would prefer it to be on disk. Write to Donald McKenzie, 49 Williams Street, Kaiapoi North Canterbury, if you have a program to suit.

USER GROUP MEETINGS

AUCKLAND:-meets 2nd Wednesday of the month, 7.30pm, Conference Rooms, Auckland Secondary Teachers College, 60 Epsom Ave, EPSOM. ph Dave 770-630 x 518(bus) or Kerry 95-355.

HAMILTON:-Waikato Tech B-block staffroom ?. Ph. Peter, Hamilton 392-500 x 897(bus) or Alison, Morrinsville 6695.

TIMARU:- Meetings to be arranged in the first week of each month. Contact Chris Stone ph. Timaru 60394.

TAURANGA:-every second Wednesday. ph. Chris & Jane 65-076.

HAWKES BAY:-Alternative months Hastings & Napier. Contact Kendall Napier 435-624, Bob Taradale 446-955, Mitch Hastings 778-235.

TAUPO: -Contact Dave 84-215.

WELLINGTON:-meets LAST Thursday of the month, 7.30pm, Royal Society Lecture Theatre, Turnbull St, Thorndon. ph. Anton 286-289 or Warren 787-005.

CHRISTCHURCH:-fortnightly workshops, Monday 6.30pm at Hagley High School. ph. Michael 582-267.

OTHER CENTRES:- let us know and we will publish details here.

THE NEWSLETTER:

CONTRIBUTIONS:-most welcome. Listings should be sent in on tape or disc.(Tape - 2 copies,one at 300 baud please). A 35c stamp gets your tape / disc returned too. Please include written explanation of listing; text only material welcome too.

DEADLINES:- For a particular issue, the last day for material is the last day of the month prior to publication.

ADVERTISING:- Rates are $20 per half page, deadline as above.

Copyright (c) THE ACORN/BBC COMPUTER USER GROUP OF NZ INC.

Published monthly except January and mailed to financial members.

MEMBERSHIP:

Membership of the users group is on payment of an annual subscription from April to March. For the 1984/5 year it is: $30.00, plus a $5.00 joining fee for new members, plus $5.00 for Auckland & Wellington members as a local branch levy, which entitles attendance at meetings as detailed above. Renewing members get all back issues of the current year; New members also get back issues to v.1 #1 Jan '84.

THE ACORN/BBC COMPUTER USER GROUP OF NZ INC
P.O.Box 9592
Wellington

Reproduced by ROBERTS PRINTING Co. Ltd, Wellington.